UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/726,976	12/03/2003	Benjamin Daniel Osecky	200300842-1	5119

22879 7590 04/08/2008

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
AHMED, ENAM	
ART UNIT	**PAPER NUMBER**
2112	

NOTIFICATION DATE	DELIVERY MODE
04/08/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
mkraft@hp.com
ipa.mail@hp.com

Office Action Summary	Application No.	Applicant(s)	
	10/726,976	OSECKY ET AL.	
	Examiner	Art Unit	
	ENAM AHMED	2112	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 January 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-36* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-36* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Non – Final

This office action is in response to applicants amendment of 24 January 2008.

Claims 1-36 remain pending.

Response to applicants arguments

Applicants arguments of 1/24/08 have been fully considered, and are not found persuasive.

Response to applicants remarks

On page 2, the applicants mentions these "portions" in Bunnel are thus not separate "computation segments", since only the third portion of memory space in Bunnel is used for "computation". The "executable code segments" referred to above are not segments of a single program, as recited in Applicants' claim 1, but only single segments from each of several possible "support and application programs".

The Examiner disagrees with the statement, and points out there are several sections in the Bunnel reference which discuss seperate segments of data or computation segments, that it would have been obvious to one of ordinary skill in the art to include separate "computation segments". Thus, the Bunnel reference teaches separate "computation segments" (column 3, lines 9-22), (column 3, lines 42-47), (column 7, lines 8-40) and (column 8, lines 58-61).

On page 3, the applicants mentions thus Bunnel does not "compil[e] source code ... to generate two code sections", as claimed by Applicants, since Bunnel clearly states that only one of the two classes of "program segments" is executable, and only executable code can constitute a "computation segment".

The Examiner disagrees with the statement, and points out that there are sections which dicuss compilation or execution of program to generate multiple code sections. Thus, the Bunnel reference teaches "compil[e] source code ... to generate two code sections" (column 7, lines 8-27).

On page 4, the applicants mention however, applicants note that the referenced section in Galpin makes no mention of "generating comparison code". Applicants assert that Galpin does not teach the use of generating comparison code to compare the results of the execution of two processes, but rather, as noted, instead only compares the states of the processes rather than the results of execution of the processes.

The Examiner disagrees with the statement, and points out the Galpin reference is not limited to only comparing the processes of the two states and can further compare the execution results. Thus, the Galpin reference teaches the use of generating comparison code to compare the results of the execution of two processes (column 2, lines 29-48), (column 2, line 62 – column 3, line 9), (column 3, lines 21-30) and (column 8, lines 28-46).

On page 4, the applicants mention however, even if, arguendo, the cited Bunnel and Galpin references contained teachings which provided each of the elements recited in Applicants' claim 1, the Examiner states a motivation which simply could not result in Applicants' invention as claimed. There is no "loose coupling" of Applicants' claimed redundant processes.

The Examiner would like to point out the motivation provided does not have to be an ideal motivation from the applicants point of view, and further a better motivation can be provided such as for control and more particularily fault-tolerance and fault-detection (column 1, lines 65-67 – Galpin reference).

On pages 5-6, the applicants mention however, where the size of transient program area is constrained, or where the access response time must be short, it is undesirable to have to duplicate instances of each program into the transient program area in order to permit execution. Thus, the present invention provides the ability to maintain the code. The above paragraph provides no indication of a computational domain comprising a time domain. It simply addresses program formatting. Thus claims 2 and 35 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out "time domain" may not explicitly be stated, however there are sections in Bunnel et al. reference which discuss time with respect to computations which would be obvious to one of ordinary skill in the art that it can be substituted with a "time domain". Hence, the Bunnel et al. reference teaches a "time domain" (column 9, lines 5-12), 9column 1, lines 29-51), (column 4, lines 26-40) and (column 4, line 49 - column 5, line 16).

On page 6, the applicants mention applicants point out that neither the term "clock cycle" nor even the word "clock" appears anywhere in the Bunnel reference. Thus claims 3, 24, and 32 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out “clock cycle” or "clock" may not explicitly be state, however there are multiple processors operating in Bunnel and it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have included “clock cycle” or "clock". Thus the Bunnel reference teaches “clock cycle” or "clock" (column 3, lines 3-18), (column 3, lines 42-47), (column 8, lines 31-36), (column 2, lines 47-63) and (column 13, lines 26-30).

On page 7, the applicants mention the above section in the de Bonet reference discusses program execution 'hanging' in either an infinite loop or executing "for an unusually long time". This paragraph bears no relation to predetermining a minimum number of processor clock cycles as a function of statistical properties of the duration of a disruptive event. Thus claims 4 and 25 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out there are operations taking place with processors and other devices. There is also indication of long time periods for execution hence it would be obvious to one of ordinary skill in the art at the time of the invention was made to incorporate predetermining a minimum number of processor clock cycles as a function of statistical properties of the duration of a disruptive event. Hence, the de Bonet reference teaches predetermining a minimum number of processor clock cycles as a function of statistical properties of the duration of a disruptive event (column 7, lines 33-42) and (column 7, lines 9-23).

On page 7, the applicants assert that there is nothing in column 3, lines 25- 48 of the Lajolo reference that addresses the subject of computational domains. Lajolo is concerned with "how [a] structure under analysis is discretized". Thus claims 5, 6, and 36 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out there are sections in Lajolo which give indications of computational domains (column 3, lines 13-48).

On page 8, the applicants mention here, Bunnel addresses loading an operating system kernel, and fails to mention anything related to the subject of executing code sections using separate resources of a processor. Thus, for at least the above reasons, claim 6 cannot be rendered obvious by the unrelated teachings.

The Examiner disagrees with the statement, and points out there are sections in the Bunnel reference which teach executing code sections using seperate or different parts of the

processor. Thus, the Bunnel reference teaches executing code sections using separate resources of a processor (column 3, lines 3-22), (column 14, lines 20-34) and (column 5, line 64 – column 6, line 9).

On page 8, the applicants assert that the Kane reference has no significant relationship (if any at all) to Applicant's claimed invention, or to claims 26 and 33, which utilize functional units and partitioned registers for the purpose of error detection, which purpose and related function is not related in any manner to "pipelining".

The Examiner disagrees with the statement, and points out the motivation for "pipelining" may not be the ideal motivation from the applicants point of view, however there are better motivations that can be provided such as however a better motivation may be provided such as parallel execution of instructions or maximum throughput (column 4, line 2 and column 11, line 58). There are sections in the Kane et al. reference which discuss parallel processors and seperate processors for different tasks. Hence, the Kane et al. reference teaches which utilize functional units and partitioned registers for the purpose of error detection (column 1, lines 28-45), (column 1, lines 55-62), (column 2, lines 7-21) and (column 7, lines 10-20).

On page 8, the applicants mention the Merkey reference is concerned with partitioning disk storage devices, and fails to even mention partitioned registers. Thus claims 9 and 28 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out the Merkey reference is not limited to partitioning disk storage devices and can be broad to include paritioning registers.

Hence, the Merkey reference teaches partitioning registers (column 13, lines 17-21), (column 13, lines 56-62), (column 16, lines 21-34) and (column 26, lines 27-37).

On page 9, the applicants mention Merkey discusses 'registering' device objects in Linux, and does not mention anything related to "executing comparison code". Thus claims 10 and 20 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out the Merkey reference does discuss "executing comparison code" (column 12, lines 42-47).

On page 9, the applicants mention applicant notes that the cited sections in Galpin discuss "sender" and "listener" processors that "synchronously step through sequential schedules". This reference is thus entirely off-point with respect to Applicants' claims 11 and 29, since Applicants' claimed system operates entirely asynchronously.

The Examiner disagrees with the statement, and points out the Galpin reference should not be limited to only asynchronous functions. There are sections in the Galpin reference which use scheduling with the processor's instructions in a variety of ways that it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have included asynchronous functions. Thus, the Galpin reference can also be used for asynchronous functions (column 1, lines 51-57), (column 4, lines 63 - column 5, line 5) and (column 6, lines 55-64).

On page 10, the applicants mention this section of the Oates reference deals with the subject of "efficiently computing a .GAMMA.-matrix", and thus bears no relationship to Applicants' claim 15, which recites "optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections". Thus claim 15 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out that the Oates reference does teach "optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections" (column 8, line 62 – column 9, line 11) and (column 78, lines 1-10).

On page 10, the applicants mention In the above-cited sections, Bunnell respectively discusses "in general, the remaining RAM memory, generally referenced by the reference numerals 50, 50' is utilized as the transient program execution area" and "memory transfer requests directed to the logical interface corresponding to the pseudo-disk device driver results in a transfer of data fully within the main memory 14, though logically consistent with a disk drive paradigm". Neither of these sections is related to using "code reorganization to dynamically translate the source code into [the] two code sections", as recited in Applicants' claim 16. Thus claim 16 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out the Bunnel reference teaches "code reorganization to dynamically translate the source code into [the] two code sections" (column 6, lines 10-20) and (column 8, lines 28-40).

On page 11, the applicants mention Brown does not teach the use of "an optimizer for modifying the output of the compiler to schedule execution of [the] redundant code. Since Brown's purpose and function are entirely different than that claimed by Applicants in claim 22, Applicants thus submit that claim 22 cannot be rendered obvious by the unrelated teaching.

The Examiner disagrees with the statement, and points out the Brown reference teaches "an optimizer for modifying the output of the compiler to schedule execution of [the] redundant code" (column 4, lines 7-27).

On page 12, the applicants mention the cited combination of references provides a "motivation for an optimizer for configuring is for an improvement in efficiency of a computer maintenance engineer". Applicants note that their claimed system (including claim 22) has absolutely nothing to do with improving the "efficiency of a computer maintenance engineer". Applicants assert that this statement so significantly mis- characterizes their claimed invention as to indicate that the cited combination of references must be inapplicable.

The Examiner would like to point out that the motivation provided does not have to be an ideal motivation from the applicants points of view, and a better motivation can be provided from Brown such as the system may be optimally deployed as a centralized maintenance system for a large computer network (column 13, lines 54-55 – Brown reference).

On page 12, the applicants assert that the Kane reference has no significant relationship (if any at all) to Applicant's claimed invention, or to claims 26 and 33, which utilize functional units and partitioned registers for the purpose of error detection, which purpose and related function is not related in any manner to "pipelining". If "pipelining" is in fact the motivation supplied by the Kane reference, then, Applicants assert that the system resulting from the cited combination of references cannot render claims 26 and 33 obvious, given the unrelated teaching and non-relevant motivation to combine.

The Examiner would like to point out that the motivation may not be an ideal motivation from the applicants point of view, however a better motivation may be provided such as parallel execution of instructions or maximum throughput (column 4, line 2 and column 11, line 58). The Kane reference does teach which utilize functional units and partitioned registers for the purpose of error detection (column 1, lines 28-45), (column 1, lines 55-62), (column 2, lines 7-21) and (column 7, lines 10-20).

35 U.S.C. 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1, 2, 3, 6, 24, 32 and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903) in view of Galpin (U.S. Patent No. 7,043,728).

With respect to claim 1, the Bunnel et al. reference teaches seperating the program into computation segments (column 3, lines 9-22), (column 3, lines 42-47), (column 7, lines 8-40) and (column 8, lines 58-61); compiling source code for at least one of the segments to generate two code sections (column 7, lines 8-27); executing each of the code sections in a different computational domain to generate respective results (column 1, lines 28-50); (column 2, lines 30-46) and (column 16, line 65 – line 17, column 10). The Bunnel et al. reference does not teach one of which is functionally redundant with respect to the other, generating comparision code for comparing results produced by the execution of the two code sections, comparing the respective results using the comparision code and executing one of the code sections to alter further flow of execution of the program only if the respective results are identical. The Galpin reference teaches one of which is functionally redundant with respect to the other (column 1, lines 34-50), (column 1, lines 58-63), (column 3, lines 21-30) and (column 6, lines 7-15); generating comparision code for comparing results produced by the execution of the two code sections (column 2, lines 29-48), (column 2, line 62 – column 3, line 9), (column 3, lines 21-30) and (column 8, lines 28-46); comparing the respective results using the comparision code and executing one of the code sections to alter further flow of execution of the program only if the respective results are identical (column 2, lines 29-48), (column 2, line 62 – column 3, line 9) and (column 3, lines 21-30). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to

incorporate one of which is functionally redundant with respect to the other, generating comparision code for comparing results produced by the execution of the two code sections, comparing the respective results using the comparision code and executing one of the code sections to alter further flow of execution of the program only if the respective results are identical. The motivation for one of which is functionally redundant with respect to the other, generating comparision code for comparing results produced by the execution of the two code sections, comparing the respective results using the comparision code and executing one of the code sections to alter further flow of execution of the program only if the respective results are identical is for control and more particularily fault-tolerance and fault-detection (column 1, lines 65-67 – Galpin reference).

With respect to claims 2 and 35, the Bunnel et al. reference teaches wherein said computational domain comprises a time domain (column 9, lines 5-12), 9column 1, lines 29-51), (column 4, lines 26-40) and (column 4, line 49 - column 5, line 16).

With respect to claims 3, 24 and 32 the Bunnel et al. reference teaches wherein the compiling step includes compiling the source code to schedule execution thereof so that a minimum number of processor clock cycles elapse between execution of a first one of the code sections and execution of the other one of the code sections (column 3, lines 3-18), (column 3, lines 42-47), (column 8, lines 31-36), (column 2, lines 47-63) and (column 13, lines 26-30).

With respect to claim 6, the Bunnel et al. reference teaches wherein said compiling step includes compiling the source code such that each of the code sections is executed using

seperate resources of the processor column 3, lines 3-22), (column 14, lines 20-34) and (column 5, line 64 – column 6, line 9).

Claims 4 and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of de Bonet (U.S. Patent No. 7,168,008).

With respect to claims 4 and 25, the Bunnel et al. reference teaches all of the limitations of claims 3 and 24. The bunnel et al. reference does not teach wherein said minimum number of processor clock cycles is predetermined as a function of statistical properties of duration of disruptive events causing said computational errors. The de Bonet reference teaches wherein said minimum number of processor clock cycles is predetermined as a function of statistical properties of duration of disruptive events causing said computational errors (column 7, lines 33-42) and (column 7, lines 9-23). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated the references Bunnel and de Bonet to incorporate wherein said minimum number of processor clock cycles is predetermined as a function of statistical properties of duration of disruptive events causing said computational errors into the claimed invention. The motivation for wherein said minimum number of processor clock cycles is predetermined as a function of statistical properties of duration of disruptive events causing said computational errors is to improve the protection system.

Claims 5 and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of Lajolo (U.S. Patent No. 6,880,112).

With respect to claims 5 and 36, the Bunnel et al. reference teaches all of the limitations of claims 1 and 34. The Bunnel et al. reference does not teach wherein said computational domain comprises a spatial domain. The Lajolo reference teaches wherein said computational domain comprises a spatial domain (column 3, lines 13-48). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Lajolo to incorporate wherein said computational domain comprises a spatial domain into the claimed invention. The motivation for wherein said computational domain comprises a spatial domain is for a high dependability level (column 5, lines 17-18).

Claims 7 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) in view of kane et al. (U.S. Patent No. 5,537,559).

With respect to claim 7, all of the limitations of claim 6 have been addressed. The Bunnel et al. reference does not teach wherein said resources comprise functional units and partitioned registers. The Kane et al. reference teaches wherein said resources comprise functional units and partitioned registers (column 1, lines 28-45), (column 1, lines 55-62), (column 2, lines 7-21) and (column 7, lines 10-20). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Kane et al. to incorporate wherein said resources comprise functional units and partitioned registers into the claimed invention. The motivation for wherein said resources

comprise functional units and partitioned registers is for parallel execution of instructions or maximum throughput (column 4, line 2 and column 11, line 58).

With respect to claim 8, all of the limitations of claim 7 have been addressed. The Bunnel et al. reference does not teach wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers. The Kane et al. reference teaches wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers (column 1, lines 18-27). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated Bunnel et al. and Kane et al. to incorporate wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers into the claimed invention. The motivation for wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers is achieving a high degree of pipelining (column 3, lines 67 – column 4, line 1 – Kane et al. reference).

Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) in view of Merkey (U.S. Patent No. 6,862,609).

With reference to claims 9, all of the limitations of claim 6 have been addressed. The Bunnel et al. reference does not teach wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the

code sections. The Merkey reference teaches wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections (column 13, lines 17-21), (column 13, lines 56-62), (column 16, lines 21-34) and (column 26, lines 27-37). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Merkey to incorporate wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections into the claimed invention. The motivation for wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections is to improve system performance.

Claims 10-14 are under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) in view of Merkey (U.S. Patent No. 6,862,609).

With respect to claims 10, all of the limitations of claim 1 have been addressed. The Bunnel et al. reference does not teach wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed. The Merkey reference teaches wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed (column 12, lines 42-47). Thus

it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated Bunnel et al. and Merkey to incorporate wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed into the claimed invention. The motivation for wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed is to improve system performance.

With respect to claim 11, all of the limitations of claim 1 have been addressed. The Bunnel et al. reference does not teach wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units. The Galpin reference teaches wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units (column 1, lines 51-57), (column 4, lines 63 - column 5, line 5) and (column 6, lines 55-64). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units into the claimed invention. The motivation for wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 12, the Bunnel et al. reference teaches including performing error handling if a discrepancy between the respective results is found (column 7, lines 29-40).

With respect to claim 13, all of the limitations of claim 12 have been addressed. The Bunnel et al. reference teaches wherein said error handling includes at least one function selected from the group consisting of re-execution (column 12, lines 36-51). The Bunnel et al. reference does not teach trapping to an error handling routing. The Galpin reference teaches failing and trapping to an error handling routine (column 7, lines 61-67). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units into the claimed invention. The motivation for wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 14, the Bunnel et al. reference teaches wherein each of the computation segments receives a set of inputs, performs at least one computation on the input values, and exposes a set of outputs to further computation (column 1, lines 29-48).

Claims 15-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of Oates (U.S. Patent No. 7,110,431).

With respect to claim 15, all of the limitations of claim 1 have been addressed. The Bunnel et al. reference does not teach the step of optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections. The Oates reference teaches the step of optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections (column 8, line 62 – column 9, line 11) and (column 78, lines 1-10). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Oates to incorporate the step of optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections into the claimed invention. The motivation for the step of optimizing one of the two code sections to execute via different registers and functional units than the other one of the code sections is to manage faults for high availbility.

With respect to claim 16, the Bunnel et al. reference teaches wherein the compiling step employs code reorganization to dynamically translate the source code into two code sections (column 6, lines 10-20) and (column 8, lines 28-40).

Claim 17 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of de Bonet (U.S. Patent No. 7,168,008).

With respect to claim 17, all of the limitations of claim 1 have been addressed. The Bunnel et al. reference does not teach wherein the step of compiling the source code is

performed by incrementally translating the source code. The de Bonet reference teaches wherein the step of compiling the source code is performed by incrementally translating the source code (column 11, lines 47-67). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Bonet to incorporate wherein the step of compiling the source code is performed by incrementally translating the source code into the claimed invention. The motivation for wherein the step of compiling the source code is performed by incrementally translating the source code is to improve the protection system.

Claims 18-19, 21 and 29-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903) in view of Galpin (U.S. Patent No. 7,043,728).

With respect to claim 18, the Bunnel et al. reference teaches seperate the program into computation segments (column 3, lines 9-22); compile source code for at least one of the computation segments to generate output (column 7, lines 8-27); each of which is configured to execute in a different computational domain (column 1, lines 28-50); (column 2, lines 30-46) and (column 16, line 65 – line 17, column 10). The Bunnel et al. reference does not teach two redundant code sections and generate comparision code for comparing respective results produced by execution of the two code sections. The Galpin reference teaches two redundant code sections other (column 1, lines 34-50), (column 1, lines 58-63), (column 3, lines 21-30) and (column 6, lines 7-15); and generate comparision code for comparing respective results produced by execution of the two code sections (column 2, lines 29-48) and (column 3, lines 21-30). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate

two redundant code sections and generate comparision code for comparing respective results produced by execution of the two code sections into the claimed invention. The motivation for two redundant code sections and generate comparision code for comparing respective results produced by execution of the two code sections is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 19, all of the limitations of claim 18 have been addressed. The Bunnel et al. reference teaches executes each of the code sections in a different computational domain to generate respective results for each of the code sections (column 1, lines 28-50); (column 2, lines 30-46) and (column 16, line 65 – line 17, column 10); peforms error handling, if a discrepancy between the respective results is found (column 7, lines 29-40). The Bunnel et al. reference does not teach compares the respective results using the comparision code. The Galpin reference teaches compares the respective results using the comparision code (column 2, lines 29-48) and (column 3, lines 21-30). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate compares the respective results using the comparision code into the claimed invention. The motivation for compares the respective results using the comparision code is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 21, all of the limitations of claim 19 have been addressed. The Bunnel et al. reference teaches wherein said error handling includes at least one function selected from the group consisting of re-execution (column 12, lines 36-51). The Bunnel et al. reference does not teach trapping to an error handling routing. The Galpin reference teaches

failing and trapping to an error handling routine (column 7, lines 61-67). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units into the claimed invention. The motivation for wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 29, all of the limitations of claim 18 have been addressed. The Bunnel et al. reference does not teach wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units. The Galpin reference teaches wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units (column 1, lines 51-57), (column 4, lines 63 - column 5, line 5) and (column 6, lines 55-64). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units into the claimed invention. The motivation for wherein the compiler uses an explicit scheduling aspect of the processor's instructions set to insure that the two code sections are each executed by a different set of functional units is for efficient

synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

With respect to claim 30, the Bunnel et al. reference teaches executes each of the code sections in a different computational domain to generate respective results for each of the code sections (column 1, lines 28-50); (column 2, lines 30-46) and (column 16, line 65 – line 17, column 10). The Bunnel et al. reference does not teach compares the repective results using the comparision code and compares the respective results using the comparision code and executes one of the code sections to alter further flow of execution of the program only if the respective results are identical. The Galpin reference teaches compares the respective results using the comparision code (column 2, lines 29-48) and (column 3, lines 21-30); executes one of the code sections to alter further flow of execution of the program only if the respective results are identical (column 2, lines 29-48), (column 2, line 62 – column 3, line 9) and (column 3, lines 21-30). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention to have combined the references Bunnel et al. and Galpin to incorporate compares the repective results using the comparision code and compares the respective results using the comparision code and executes one of the code sections to alter further flow of execution of the program only if the respective results are identical into the claimed invention. The motivation for compares the repective results using the comparision code and compares the respective results using the comparision code and executes one of the code sections to alter further flow of execution of the program only if the respective results are identical is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

Claim 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) in view of Merkey (U.S. Patent No. 6,862,609).

With respect to claim 20, all of the limitations of claim 19 have been addressed. The Bunnel et al. reference does not teach wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed. The Merkey reference teaches wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed (column 12, lines 42-47). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated Bunnel et al. and Merkey to incorporate wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed into the claimed invention. The motivation for wherein the respective results are compared by executing the comparision code in a different computational domain from the domain in which one of the code sections was executed is to improve system performance.

Claim 22 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of Brown (U.S. Patent No. 6,446,058).

With respect to claim 22, all of the limitations of claim 18 have been addressed. The Bunnel et al. reference teaches to schedule execution of the redundant code sections so that a minimum number of clock cycles elapse between execution of a first one of the sections and execution of the other one of the code sections (column 3, lines 3-18), (column 3, lines 42-47) and (column 8, lines 21–36). The Bunnel et al. reference does not teach an optimizer for modifying the output of the compiler. The Brown reference teaches an optimizer for modifying the output of the compiler (column 4, lines 7-27). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Brown to have incorporated an optimizer for modifying the output of the compiler into the claimed invention. The motivation for an optimizer for modifying the output of the compiler is so the system may be optimally deployed as a centralized maintenance system for a large computer network (column 13, lines 54-55 – Brown reference).

Claim 23 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728), Brown (U.S. Patent No. 6,446,058) further in view of Oates (U.S. Patent No. 7,110,431).

With respect to claim 23, all of the limitations of claim 18 have been addressed. The Bunnel et al. reference does not teach an optimizer for configuring one of the redundant code sections to execute via different registers and functional units than the other one of the code sections. The Brown reference teaches an optimizer for configuring (column 4, lines 7-27). The Oates reference teaches one of the redundant code sections to execute via different registers and functional units than the other one of the code sections (column 8, line 62 – column 9, line 11). Thus it would have been obvious to one of ordinary skill in the art at the time of the

invention to combine Bunnel et al. and Brown to incorporate an optimizer for configuring into the claimed invention. It would also have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the references Bunnel et al. and Oates to incorporate one of the redundant code sections to execute via different registers and functional units than the other one of the code sections into the claimed invention. The motivation for an optimizer for configuring is for parallel execution of instructions or maximum throughput (column 4, line 2 and column 11, line 58). The motivation for one of the redundant code sections to execute via different registers and functional units than the other one of the code sections is to manage faults for high availbility.

Claims 26 and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) further in view of kane et al. (U.S. Patent No. 5,537,559).

With respect to claims 26 and 33, all of the limitations of claim 18 and 31 have been addressed. The Bunnel et al. reference teaches wherein the compiler compiles the source code such that each of the code sections is executed (column 3, lines 3-22) and (column 14, lines 20-34). The Bunnel et al. reference does not teach using a different set of functional units and partitioned registers of the processor. The Kane et al. reference teaches using a different set of functional units and partitioned registers of the processor (column 1, lines 28-45), (column 1, lines 55-62) and (column 2, lines 7-21). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Kane et al. to incorporate using a different set of functional units and partitioned registers of the processor into the claimed invention. The motivation for using a different set of functional

units and partitioned registers of the processor is achieving a high degree of pipelining (column 3, lines 67 – column 4, line 1 – Kane et al. reference).

Claim 27 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728) in view of kane et al. (U.S. Patent No. 5,537,559).

With respect to claim 27, all of the limitations of claim 26 have been addressed. The Bunnel et al. reference does not teach wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers. The Kane et al. reference teaches wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers (column 1, lines 18-27). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated Bunnel et al. and Kane et al. to incorporate wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers into the claimed invention. The motivation for wherein the partitioned registers are used to effect the detection and repair of errors in the registers and paths to/from the registers is achieving a high degree of pipelining (column 3, lines 67 – column 4, line 1 – Kane et al. reference).

Claims 28 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728), kane et al. (U.S. Patent No. 5,537,559) in view of Merkey (U.S. Patent No. 6,862,609).

With respect to claim 28, all of the limitations of claim 26 have been addressed. The Bunnel et al. reference does not teach wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections. The Merkey reference teaches wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections (column 13, lines 17-21), (column 13, lines 56-62), (column 16, lines 21-34) and (column 26, lines 27-37). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Merkey to incorporate wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections into the claimed invention. The motivation for wherein the partitioned registers are utilized by encoding register names from a first set of registers into instructions in a first one of the code sections and encoding register names from a second set of registers into instructions in the other one of the code sections is to improve system performance.

Claims 31 and 34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bunnel et al. (U.S. Patent No. 5,594,903), Galpin (U.S. Patent No. 7,043,728).

With respect to claims 31 and 34, the Bunnel et al. reference teaches means for compiling source code for at least part of the program to generate two code sections (column 7, lines 8-27); wherein each of the code sections is executed in a different computational domain to generate respective results (column 1, lines 28-50); (column 2, lines 30-46) and (column 16, line 65 – line 17, column 10); means for performing error handling, if a discrepancy between the respective results is found (column 7, lines 29-40). The Bunnel et al. reference does not teach one of which is functionally redundant with respect to the other, means for generating comparision code for comparing results produced by execution of the two code sections, means for comparing the respective results using the comparision code. The Galpin reference teaches one of which is functionally redundant with respect to the other (column 1, lines 34-50), (column 1, lines 58-63), (column 3, lines 21-30) and (column 6, lines 7-15); means for generating comparision code for comparing results produced by execution of the two code sections (column 2, lines 29-48) and (column 3, lines 21-30); means for comparing the respective results using the comparision code (column 2, lines 29-48), (column 2, line 62 – column 3, line 9) and (column 3, lines 21-30). Thus it would have been obvious to one of ordinary skill in the art at the time of the invention was made to have combined the references Bunnel et al. and Galpin to incorporate one of which is functionally redundant with respect to the other, means for generating comparision code for comparing results produced by execution of the two code sections and means for comparing the respective results using the comparision code into the claimed invention. The motivation for one of which is functionally redundant with respect to the other, means for generating comparision code for comparing results produced by execution of

the two code sections and means for comparing the respective results using the comparision code is for efficient synchronism, fault-detection, fault-tolerance while effectuating loose coupling of the processes (column 2, lines 46-48).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Enam Ahmed whose telephone number is 571-270-1729. The examiner can normally be reached on Mon-Fri from 8:30 A.M. to 5:30 P.M.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Jacques Louis-Jacques, can be reached on 571-272-6962.

The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

EA

9/26/07

/JACQUES H LOUIS-JACQUES/

Supervisory Patent Examiner, Art Unit 2112